Addressing teachers' concerns about the Prog&Play serious game with context adaptation

Mathieu Muratet, Elisabeth Delozanne, Patrice Torguet, Fabienne Viallet

Addressing teachers' concerns about the Prog&Play Serious Game with context adaptation

Mathieu Muratet

INS HEA, 58-60, avenue des landes,
92150 Suresnes, France
E-mail: mathieu.muratet@inshea.fr

Elisabeth Delozanne

LIP6, Université Pierre et Marie Curie, 4 place Jussieu,
75005 Paris, France
E-mail: elisabeth.delozanne@lip6.fr

Patrice Torguet

IRIT, Université Paul Sabatier, 118 route de Narbone,
31400 Toulouse, France
E-mail: patrice.torguet@irit.fr

Fabienne Viallet

UMR EFTS, Université Paul Sabatier, 118 route de Narbone,
31400 Toulouse, France
E-mail: fabienne.viallet@univ-tlse3.fr

Abstract: Computers games are supposed to provide a good learning environment. We study a large-scale use (260 students and 20 teachers) of Prog&Play, a serious game specially designed to teach computer programming fundamentals. In earlier quantitative studies, we found, through a students' motivation survey, that the students' interest for Prog&Play was not only related to the intrinsic game quality, it was also related to the teaching context. In this paper, we investigate contextual effects that influence motivational and learning benefits while using Prog&Play in different teaching settings. Results from a qualitative study of 41 students' and 18 teachers' comments, suggest guidelines for a successful implementation of Prog&Play.

Keywords: Serious games; teaching programming; motivation; usage feedback.

Reference to this paper should be made as follows: Muratet, M. and Delozanne, E. and Torguet, P. and Viallet, F. (2012) 'Addressing teachers concerns about the Prog&Play Serious Game with context adaptation', *Int. J. Learning Technology*, Vol. 7, No. 4, pp.419–433. *The definitive version is available at http://www.inderscience.com/.*

Biographical notes: Mathieu Muratet received his PhD in Computer Science in 2010 from the University of Toulouse, France. He is an assistant professor in the INS HEA, France. His research interests include serious games, teaching programming, motivation and usage feedback.

Elisabeth Delozanne is associate professor in Computer Science at Université Pierre et Marie Curie, Paris, France. Her research interest is in congitive diagnosis, Intelligent tutoring systems, e-learning and teaching progamming.

Patrice Torguet is assistant professor in Computer Science at IRIT, Université de Toulouse, Toulouse, France. His research interests include networked virtual environments, massively multiuser virtual environments and serious games.

Fabienne Viallet is assistant professor in Computer Science at Université Paul Sabatier, Toulouse, France. Her research interest is in science education, serious game and teaching progamming.

1 Introduction

ForZyda (2005), a serious game is "*a mental contest, played with a computer in accordance with specific rules, that uses entertainment to further government or corporate training, education, health, public policy, and strategic communication objectives.*" Serious games are widely studied in the literature for their motivation effects (Bizzocchi and Paras, 2005), learning benefits (Johnson and Wu, 2008; Hartness, 2004) and entertainment potential (Rodrigo et al., 2008). Other papers deal with tools and models to design serious games (Marfisi-Schottman et al., 2010; Yessad et al., 2010) or to evaluate player skills (Thomas et al., 2011). The work presented here is a contribution to that field of research: we investigate contextual effects that influence motivational and learning benefits of serious games.

Our project, called Prog&Play, aims to increase students' motivation for programming through learning and playing a serious game. Our basic assumptions are (i) that video games are exciting for students, and (ii) that they can provide also a good context in which to embed computer programming teaching materials. In this paper, we investigate which conditions benefit such a game to the teaching and learning of programming for beginners. In the first section, we discuss background and related work. In the second section, we present the Prog&Play system, and then the different experiments we conducted to test it. Finally, we analyze students' and teachers' comments to develop guidelines for a successful implementation of Prog&Play.

2 Background and related work

Our work is based on learning sciences research findings concerning the basics of programming and serious gaming.

2.1 Programming basics

We began by reviewing different curricula for teaching algorithms and programming to beginners. The main topics taught in Programming Fundamentals (ACM and IEEE-CS, 2008, p. 38) are basic syntax and semantics of a higher-level language, variables, types, expressions, assignments, simple input/output, conditional and iterative control structures, functions, parameter passing and structured decomposition. The learning objectives are: to analyze and explain the behavior of simple programs involving the fundamental programming constructs; to modify and expand short programs that use standard conditional and iterative control structures and functions; to design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, and the definition of functions; to choose appropriate conditional and iteration constructs for a given programming task; to apply the techniques of structured (functional) decomposition to break a program into smaller pieces; and to describe the mechanics of parameter passing.

These topics are taught through lectures, exercises, practice sessions and sometimes by personal or collaborative programming projects that involve more complex problems. The introductory programming course is generally assessed by a written exam, often accompanied by several programming tasks that are distributed throughout the course (McCracken et al., 2001). A considerable amount of research has been done to study the ineffectiveness of different teaching strategies, to understand the cognitive difficulties encountered by students when they start learning programming (Du Boulay, 1989; Jenkins, 2002), or to analyze the reasons why so many students drop out or fail to pass the exams (Kinnunen and Malmi, 2006). There is no single answer to these problems.

With respect to teachers, Pears et al. (2007) identify many different factors that can influence students' success: the content the teacher chooses to emphasize (problem solving, a particular language, or system or code production environment), language and paradigm choice (functional, imperative, or object-oriented), and the different tools designed to support teaching, learning and assessment of introductory programming (visualization tools, automated assessment tools, programming environments, micro-world, intelligent tutoring systems).

On the students' side, few students find learning programming easy. Jenkins (2002) identified several different issues centered on the nature of programming and on the ways it is taught and learnt. He especially stressed the lack of interest of lectures covering details of syntax and exercises that involve simple mathematical manipulations of collections of students marks, stock levels, sport statistics or bank account details. Many researchers advocate that "*an effective approach is to encourage learners to work immediately on meaningful, realistic tasks*" (Greitzer et al., 2007). Tools like Scratch (Maloney et al., 2004) or Alice2 (Kelleher et al., 2002) are used to make first programming experiences more engaging.

Our basic ideas in the Prog&Play project are (i) to anchor learning into game-based problem solving situations appealing to students (ii) to provide teachers with a tool that leaves them free to choose the teaching strategy adapted to their teaching context, particularly the choice of language and paradigm.

2.2 Video game approach

Motivation is a main feature of video games. It maintains a player's involvement to achieve fixed game objectives. In a serious game context, Garris et al. (2002) identify a tacit model of learning in most studies of instructional games. First, the objective is to design an instructional program that incorporates certain features or characteristics of games. Second, these features trigger a cycle that includes user judgments or reactions such as enjoyment or interest, user behaviors, and further system feedback. The game cycle is a defining characteristic of computer gameplay. Those are the features that training professionals hope to capture and incorporate in instructional applications. Another feature widely used by video games is storytelling. It enables to create a logical universe where the player is the main actor in a sequence of events. Storytelling may be used to increase students' motivation and students' involvement (Lane et al., 2010; Lee et al., 2010; Rowe et al., 2010). Results of experimental studies designed to examine the effectiveness of serious games and video games on players' learning and engagement is reviewed by Girard et al. (2012).

A very popular use of the game based learning approach to teach programming is asking students to implement their own video game. Chen and Cheng (2007) use C++, through a collaborative project, to enable students to build a small-to-medium scale interactive computer game in one semester, using a game framework. Gestwicki and Sun (2008) based a case study on EEClone. This game is an arcade-style computer game implemented in Java. Students analyse various design patterns within EEClone, and from this, learn how to apply design patterns in their own game software.

Colobot (2011) is the only example we know of a video game to teach programming that mixes interactivity, storytelling and programming. In this game, the user must colonise a planet using robots that s/he programs in a specific object-oriented language similar to C++.

Other serious games are based on competition. Robocode (Hartness, 2004) is a Java programming game, where the goal is to develop a robot tank to fight against other tanks programmed by other players. Robot battles are running real time and on-screen. It is suitable for all kinds of programmers from beginners (a simple robot behaviour can be written in just a few minutes) to experts (perfecting an AI - Artificial Intelligence - can take months). Other such games are Marvin's Arena (2011) using any .NET compatible language, GunTactyx (2011) using SMALL and Robot Battle (2011) using a specific script language.

In the Prog&Play project, we use a storytelling approach where students have to carry out missions similar to Colobots, but it is also possible to organize competitions between students' programs. In this paper, we focus on the scenario based approach which progressively embeds the pedagogical objectives. The approach taken to structure the learning situation relies on four basic principles: (i) the learner reads the game objective, (ii) s/he defines a strategy to achieve this objective, (iii) then s/he programs the game units with simple programming constructs using the Prog&Play library (iv) finally s/he sees the results of his/her program in the game environment.

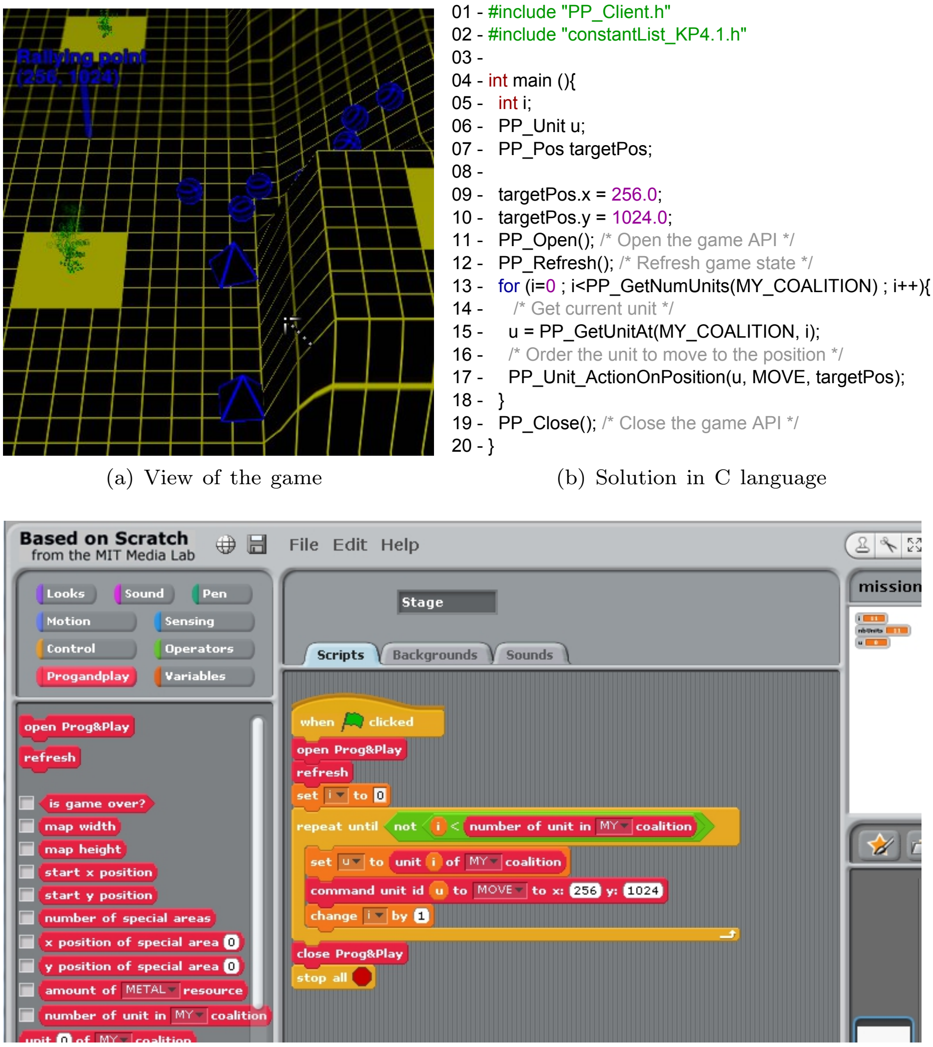
3 Project design and implementation

From a large survey with 900 undergraduate students in computer science (780 male and 170 female), we learnt that real-time strategy (RTS) is one of the most popular game type in our target public (Muratet et al., 2009). So, Prog&Play is based on an open source simplified real-time strategy game called Kernel Panic (2011). Kernel Panic uses computer science metaphors: it manipulates bits, pointers and assemblers as units. Resources are limited to time and space, and the technology improvement tree only has 10 units. All units are free to create and low-end vectorial graphics match the game universe. These characteristics emphasize strategy and tactics in an action-oriented game while remaining user-friendly.

In RTSes, a player gives orders to his/her units to carry out operations (i.e. moving, building, and so forth). Typically, these instructions are given by clicking on a map with the mouse. We modified the game to allow the player to give these instructions through a program. Students interact with the game using the Prog&Play Applicative Programming Interface (API). This API simplifies programming as much as possible. It hides the game synchronisation complexity and gives access to game data like unit features (e.g. number, position, type), map size, etc. Using these data, student's program sends commands to the game. When the game receives these commands, it executes them, modifying the game state.

It was an important design decision to leave teachers free to choose the Interactive Development Environments (IDE) and the programming language. Several versions of this API are available in the most popular programming languages used to teach beginners: Ada, C, Compalgo, Java, OCaml, and Scratch.

To map learning objectives into the game, we chose to use a story-based approach. We offered students the following scenario: "*For a number of years, a secret war has been rife inside computers. Steady attacks have been led against innocent victims. Today is your turn. Your aggressor captured your mouse controller. You must recover it. Your only solution: programming*". To achieve this final objective, we divided the scenario into several progressive missions. The first missions illustrate specific programming concepts and allow students to discover the Prog&Play library smoothly. For instance, the fourth mission aims at introducing iterative control structure to loop through each unit and move them toward the right position. Following quote is the in game briefing of the fourth mission: "*All units you control are weakened. You must repair them before starting a counter attack. The latest Assembler available to repair your units is moving to the rallying point (256, 1024). Move all units to this position*". Figure 1 presents a screenshot of the fourth mission (1a) and a solution written in C language (1b) and Scratch language (1c). In the last mission, students use all the programming concepts they have learnt to define and implement a winning strategy. Then they test their strategies against the computer or against other students programs.



(a) View of the game

```
#include "PP_Client.h"
#include "constantList_KP4.1.h"

int main (){
  int i;
  PP_Unit u;
  PP_Pos targetPos;

  targetPos.x = 256.0;
  targetPos.y = 1024.0;
  PP_Open(); /* Open the game API */
  PP_Refresh(); /* Refresh game state */
  for (i=0 ; i<PP_GetNumUnits(MY_COALITION) ; i++){
     /* Get current unit */
    u = PP_GetUnitAt(MY_COALITION, i);
    /* Order the unit to move to the position */
    PP_Unit_ActionOnPosition(u, MOVE, targetPos);
  }
  PP_Close(); /* Close the game API */
}
```

(b) Solution in C language

```
when clicked
open Prog&Play
refresh
set i to 0
repeat until not i < number of unit in MY coalition
  set u to unit i of MY coalition
  command unit id u to MOVE to x: 256 y: 1024
  change i by 1
close Prog&Play
stop all
```

(c) Solution in Scratch language

Figure 1: Fourth mission of Prog&Play dealing with loop.

4 Research questions

In this study we aim to get more information about contextual effects in order to benefit from a serious game. Earlier quantitative studies (Muratet et al., 2011, 2012) suggested that workshop or project contexts of Prog&Play teaching settings were more beneficial for learners' satisfaction than regular lectures along with practice sessions. Our research questions are as follows:

1. What features of the game were stressed by teachers to explain Prog&Play adoption or non-reuse?

2. What features of the teaching context promote learning with Prog&Play?

4.1 Usage settings and participants

We studied the usage of Prog&Play in seven different settings (S1 to S7) involving 258 students and 20 teachers (table 1). The subjects were aged around 19 years and 16% of students were female. Teachers organized the pace, schedule and evaluation of students work with respect to their institutional constraints. No member of the Prog&Play design team was involved as a teacher in settings 4, 6 and 7. In setting 1, 3 and 6, students were volunteers but in the other settings, it was mandatory to use Prog&Play in practice sessions. In settings 4 and 5, Prog&Play practice sessions were mandatory and integrated in the regular course, while in the other settings it was used in addition to the regular course. In settings 6 and 7, the two teachers especially designed courses called "To learn with Information Technology" and "To learn differently" to investigate new pedagogical approaches with Prog&Play in two different universities. In settings 6 and 7 a special evaluation was set up including the quality of programming in the Prog&Play context.

Teachers in settings 6 and 7 continue using Prog&Play every year as a context to anchor programming concepts in a project oriented course they designed on a game-based pedagogical approach. In setting 4, after a first use of Prog&Play in a regular course, teachers are now using it as a remedial session for low performing students. In the other settings, Prog&Play was used once but the experience stopped in the following years.

In every setting, Prog&Play was already installed on the computers and a teacher was in the room and provided help when asked by students playing with Prog&Play. Only in the 6th setting, after 5 sessions with a teacher, students had to complete the game at home with e-mail based support to install the game and the programming environment or to debug their programs.

4.2 Materials and method

On the students' side, questionnaires were used to collect information on their own motivation. The questionnaire was made up of three parts. The first part asked questions about students' attitude towards programming and gaming. The second part dealt with students' satisfaction. The answers were collected using seven Likert Items. The last part asked for free comments and suggestions. On the teachers' side, we delivered a questionnaire and conducted informal interviews just after the end of a unit using Prog&Play. We also investigate if teachers went on using Prog&Play in their next programming units (S4, S6, S7).

In previous works (Muratet et al., 2012), a quantitative study showed that the students' interest for Prog&Play was not only related to the intrinsic game quality, it was also related to the teaching context (Figure 2). In this paper, we assume, that, from teachers' point of view, an objective criteria of Prog&Play successful use is whether teachers reuse Prog&Play after a first experimental use. To understand the reasons of adoption or non-reuse, we investigate what features of the games were stressed by teachers in their free comments (N=18) and during the interviews (N=6). We also look into students' free comments (N=41) to support our analysis.

Table 1 Usage of Prog&Play in different settings.

	Institution	**Year**	**Teachers; Participants / registred students**	**Language and Teaching contexts**	**Time spent on game**
1	Tech. college A	2008	T: 2; S: 15/196	Compalgo, Windows, Volunteer students, Workshop apart from regular teaching	5 * 1h30
2	Tech. college B	2009	T: 2; S: 35/115	C, Windows, Practice for unsuccessful students in addition to regular teaching	3 * 1h30
3	Tech. college B	2009	T: 2; S: 16/108	Java, Windows, Volunteer students, Workshop apart from regular teaching	3 * 1h30
4	Tech. college C	2009	T: 2; 60/60	C, Linux, Compulsory practice sessions for every student	5 * 1h30
		2011	T: 1; 10/60	C, Linux, Remedial sessions for low performing students	8 * 1h
5	University A	2009	T: 15; S: 99/198	OCaml, Linux, Compulsory practice sessions for half the students	2 * 2h
6	University B	2009	T: 1; S: 9/??	C, Windows, Volunteer students, Workshop part of a regular IT course	6 * 2h + homework
		2010			
		2011	T: 1; S: 13/??		
7	Tech. college D	2009	T: 1; S: 15/15	C, Windows, Workshop part of regular teaching especially designed for failing students	5 * 2h
		2010			
		2011	T: 1; S: 30/30		

Comments and interviews were collected in French. In the following, teachers' and students' quotations are translations. From this analysis, we suggest guidelines for a successful teaching implementation of Prog&Play.

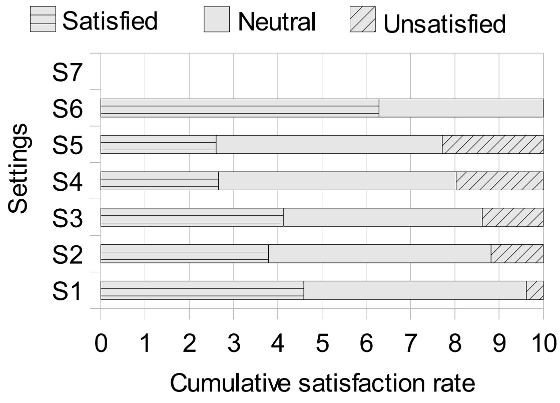


Figure 2: Global students' satisfaction from each setting.

5 Results, analysis and discussion

Comments by students and teachers from settings 1, 6 and 7 are nearly unanimously enthusiastic about Prog&Play. Both pointed out qualities of the game to entice students to **spend more time programming** by providing them with an **enjoyable experience** to develop their **creativity**. Examples of comments about time and enjoyment are:

> "*It works like a charm !!! Students were actually involved: they worked at home even though I didn't ask them*", setting 7, teacher 1.
>
> "*It was a surprise for me and for the students, to see the large diversity of the strategies that students implemented to solve the missions*", setting 6, teacher 1.
>
> "*This project let me discover programming in a domain that I enjoy very much: video games. It increases my motivation for programming. Indeed programming invokes imagination. In this unit, the assignment was exactly that. We were asked to imagine a strategy to defeat the opponent. And, in my opinion that was great*", setting 6, student 5.

They noticed that the serious game is also a good solution to **contextualise programming** exercises.

> "*Students said that they enjoyed to write "a real program" and not only a mathematical function*", setting 6, teacher 1.
>
> "*This workshop should be integrated into regular teaching in our technological college. It is a concrete teaching where we understand that our algorithms serve a useful purpose*", setting 1, student 2.

Only a few comments (from teachers) point out some difficulties to install the game on their computers.

Comments from settings 2, 3 and 4 are more diverse. Lots of comments corroborate those described previously. Teachers expressed that they **underestimated the preparation time** to plan Prog&Play sessions. Some students expressed a **lack of teachers' guidance** and a **lack of time** to complete missions.

> "*Without being already familiar with the game, it's difficult to explain students how to program the strategies they are thinking about*", setting 2, teacher 1.
>
> "*In order to better prepare this, we would have needed to adapt the game more closely to the knowledge and skills of students (simplifying libraries), the schedule and the assessment*", setting 4, teacher 1.
>
> "*The only point I am disappointed with is that we didn't get enough support from the teacher. I would prefer to go slower with the teacher paying more attention to my concerns*" setting 2, student 1.

In setting 5, comments are ambivalent. If teachers agreed on the motivation aspect of Prog&Play, they are sceptical about the efficiency of Prog&Play sessions. Some of them stated that they were counter-productive for several students. They invoked a main reason: **paper documentation** was tedious. Students' main criticisms also adressed these sessions paper documentation.

> "*It's a great idea to combine programming with a video game, but I think that explanations weren't clear enough and had a fatal effect on the motivation to "program while playing"*", setting 5, student 12.

Some teachers commented on **game exposure duration**. For two of them, Prog&Play was a loss of time for students, and two teachers advocated more time to benefit from the game.

> "*Missions were too long to be completed and, consequently, very few exercises were completed in a session*", setting 5, teacher 5.

> "*In my opinion, it is necessary to put more game time to make a coherent course and not to remain an alibi to be more appealing*", setting 5, teacher 11.

Teachers stated some students' cognitive overload due to the increased complexity to **manage the programming environment**, the program to be written and the game. For example: to switch from the programming environment to Kernel Panic is not straight forward for some students and may add complexity.

> "*What was a problem for students was managing the game environment. It is well done, and I don't know how to improve it but it is just an overhead, inherent to the game context. In my opinion, it is worth a try using Prog&Play next year in the hope that, at least teachers will be more familiar with the game environment*", setting 5, teacher 6.

One teacher and some students remarked that Prog&Play was not congruent with a functional paradigm: controlling actions on units was more adequate in an imperative or object-oriented paradigm than in a functional paradigm.

> "*In my opinion, Prog&Play is a good idea, but the Prog&Play sessions were too early: it requires students to use concepts that they do not master yet (using functions and variables from a library). The Caml language does not seem to be appropriated to Prog&Play use. Maybe it would be more interesting to introduce this game in the second semester to go with the C language*", setting 5, teacher 2.

> "*Using a video game to learn programming is a very good idea but the Caml language is not really the most suitable solution to program a video game*", setting 5, student 23.

5.1 Analysis and discussion

These experiments suggest that introducing Prog&Play to teach the basics of programming languages requires teacher's involvement and a special organisation for Prog&Play sessions involving five main organisational factors: content taught, game exposure duration, guidance, evaluation and meta-cognition.

5.1.1 Teacher's involvement

We noticed that the criterion of success (Prog&Play reuse after a first experiment) is clearly related to teacher's involvement: in setting 4, 6 and 7, the teachers took the initiative to implement the game in their teaching and were responsible for all the technical aspects to install the game on computers in their institutions. First, this shows that Prog&Play could be implemented into sessions without the Prog&Play team management. Second, comments collected in settings 6 and 7 are much better than settings where Prog&Play team manages implementations. This is due to teachers' involvement to build teaching around the serious game. In other settings, we asked teachers to experiment the game and the Prog&Play team was responsible for solving all technical problems. Few teachers have taken part in teaching Prog&Play implementation and the others have supported experimentation but was not really invested in. In consèquence, when experimentations has been ended, any teachers was able to pursue initiative. The Prog&Play teaching renewal is not only linked on success or failure in implementation but also on teachers' motivation to experiment new pedagogical tools. The role of teacher involvement is well documented in pedagogical literature (Egenfeldt-Nielsen, 2006).

5.1.2 Content

The contents taught in regular settings versus in the game are not exactly the same. Prog&Play missions aim to anchor programming concepts in a context that students find realistic.

> "*Students appreciate playing the game and seeing that their programs have a true effect on the game*", setting 5, teacher 6.

> "*It was easier for me to build programs in a game. Running them was more interesting*", setting 1, student 1.

Explicit concepts are similar (assignment, conditional and repetitive control structures as well as functions), but they are applied in different contexts: on the one hand, using simple mathematical exercises in regular courses, and on the other hand, in Prog&Play, commanding units in a world that evolves with time (while mathematical numbers are static). If we agree that learning is contextual, programming in a dynamic world is a big change for students. In regular course, mathematical concepts are supposed to be known by students, while managing the game environment is a context to be taught for non player-students.

An important implicit skill in Prog&Play use is browsing and using a library and usually, this is not taught to beginners. Teachers in setting 5 quoted this as a negative feature, but teachers in project settings quoted this as the most positive feature. This relates tightly to the course objectives: to introduce functions (regular course) or to use functions in a realistic context (project setting).

> "*Using Prog&Play involves concepts that students do not master yet, for instance using functions and variables defined in a library*", setting 5, teacher 4.

> "*Prog&Play introduces students to programming in the real world where they will have to program while browsing a library of predefined functions; they can see how important it is to comment functions and parameters*", setting 6, teacher 1.

5.1.3 Game exposure duration

This factor is related to the game based aspect. Players like to discover the game world, the different units and their properties and possibilities. So, in Prog&Play sessions, the time spent to explore the game world is important to understand how the game world reacts to student's programs.

In a regular course, each exercise is directly linked to a single concept to be taught. To integrate the game in a regular setting, teachers broke the different missions into different sessions to fit into the teaching schedule. So low performing students were prompted by their teachers to complete missions in the allotted time, in a directive approach and not in a discovery based approach. But this decomposition and the stipulated schedule go against the student's involvement in the game where the missions were designed to be in a continuous flow (Csikszentmihalyi (1991) defines flow as a state in which a game player loses track of time and is absorbed in the experience of game play). Workshops or projects allow teachers to setup a more flexible organisation coherent with the game flow where students may discover the game at their own pace (*intrinsic* motivation):

> "*The solution of the seventh mission took a long time to be achieved. Lots of ideas have been considered and left unused. At the end, hundreds of code lines have been written. I saw my army destroyed many many times. But, each attempt brought me closer to victory and kept me in suspense. Due to this suspense I completed this mission*", setting 6, student 5.

5.1.4 Guidance

Game exposure alone is not completely adequate without teacher mediation. This mediation implies clarification of programming concepts, construction of links between Prog&Play programs and non-Prog&Play programs, provision of some structures for Prog&Play tasks and explorations. "*Structure doesn't imply authoritarianism*" (Clements and Meredith, 1993). It is difficult to balance guidance according to students' skills. Some high performing students like to find solutions by themselves while low performing students need to be prompted and guided.

> "*What can we do when students don't know how to read an exercise prescription? When the text is short, the teacher can reword and give examples. What can we do with a long documentation like the one we wrote for the Prog&Play Caml sessions? The teacher has to extract pertinent information for students. As far as examples are concerned, it's difficult to find them. [...] It is necessary to simplify and review assignments and documentation*", setting 5, teacher 3.

> "*The strength and weakness of this project are that it requires much initiative and patience. Strength, because a student will benefit from this experience if he doesn't drop out and makes numerous trials. Weakness, because a student*

wouldn't learn anything if he didn't know how to take initiatives, and if he didn't know how to break deadlocks. [...] I wish more units could teach us to develop analysis and inquiry competence like in this Prog&Play unit", setting 6, student 6.

5.1.5 Evaluation and Meta-cognition

The fourth factor is related to evaluation. Unfortunately, questionnaires and interviews were collected before students' evaluation. None of them deals with that factor. But, in our opinion, it is a crucial point to analyze the teaching context. Students want to pass the course, so they are highly motivated by the evaluation process (*extrinsic* motivation). It is clear that, when the final evaluation does not take into account the Prog&Play programming context, efforts to overcome the difficulty to win the missions seem to be a loss of time for students and teachers. So it is not surprising that students were less motivated in settings where the evaluation did not take into account Prog&Play features (S4, S5), neutral when there was no evaluation (S1, S2, S3) and satisfied when the evaluation was based on Prog&Play use (S6, S7).

The fifth factor is to be investigated more: the meta-cognitive and reflexive factor. In fact the two project settings were clearly labeled as new learning approaches and students were asked to reflect on what they had learnt during the game-based course. Which leads to our last hypothesis that this reflexive phase could be a vital factor of success (Kiili, 2007).

6 Conclusion

In this paper, we have presented Prog&Play, a game-based learning environment and data collected when it was introduced as practical exercises or project sessions in different university settings involving 260 students and 20 teachers. We investigated two research questions:

1. What features of the game were stressed by teachers to explain Prog&Play adoption or non-reuse?
2. What features of the teaching context promote learning with Prog&Play?

The questionnaires collected from students and the different feedback given by teachers show that there is a clear influence of the teaching setting on Prog&Play adoption or non-reuse. Our hypotheses are that Prog&Play is better suited to a project setting with time to discover the game and environment along with an adapted evaluation process including a reflection phase. These conditions improve both *extrinsic* and *intrinsic* motivation. *Extrinsic* motivation is influenced by evaluation and teachers' involvement whereas *intrinsic* motivation depends on game quality and exposure duration.

This study suggests some guidelines to promote learning with Prog&Play:

- providing time to teachers to prepare the game sessions, to master the game environment and to understand pedagogical changes induced by a serious game approach;

- giving time to students to discover the game world and rules;
- clearly stating the pedagogical objectives of the game-based sessions and of learning outcomes that will be evaluated to pass the course;
- balancing guidance to support autonomous students as well as low confident students;
- encouraging peer collaboration;
- explicit teaching of implicit skills: why is it important to use predefined functions and variables; how to choose and use them browsing a large library of functions; how to work in a new programming environment; how to deal with input/output data in the game world and to switch between the game and the program;
- managing a reflexive phase about what has been learnt during the game-based sessions;
- set up an evaluation that takes into account the skills worked with the serious game to assess aimed knowledge.

The data collected show that, using a serious game only as an illustration tool inside regular teaching seems to limit the possibilities of learning and motivation. The success depends to a large extent on the depth of their Prog&Play experience and the instructional support given to them. Our future work will focus on assessment of these guidelines with other serious games. Prog&Play and associated resources are freely distributed at `http://www.irit.fr/ProgAndPlay/index_en.php`.

Acknowledgments

We thank John Wisdom and Thomas Joufflineau for helping with the English writing, and teachers from Universities and IUT of Toulouse and Paris who used Prog&Play in their courses.

References

ACM, IEEE-CS: Computer Science Curriculum 2008: An Interim Revision of CS2001. ACM Press. and IEEE Computer Society Press., New York, (2008)

Bizzocchi, J., Paras, B.: Game, Motivation, and Effective Learning: An Integrated Model for Educational Game Design. In: International DiGRA Conference, (2005)

Chen, W.-K., Cheng, Y. C.: Teaching Object-Oriented Programming Laboratory With Computer Game Programming. In: Education, IEEE Transactions on, 50(3), pp. 197–203, (2007)

Clements, D. H., Meredith, J. S.: Research on Logo: Effects and Efficacy. In: Journal of Computing in Childhood Education, 4, pp. 263–290, (1993)

Colobot, `http://www.ceebot.com/colobot/index-e.php`, accessed 19 January 2011

Csikszentmihalyi, M.: Flow: The psychology of optimal experience. Reprint edition, New York: Harper Perennial, (1991)

Du Boulay, B.: Some Difficulties of Learning to Program. In: Journal of Educational Computing Research, 2(1), pp. 57–73, (1986), Reprinted as du Boulay (1989)

Egenfeldt-Nielsen, S.: Overview of research on the educational use of video games. In: Digital Kompetanse, 1(3), pp. 184–213, (2006)

Garris, R., Ahlers, R., Driskell, J. E.: Games, Motivation, and Learning: A Research and Practice Model. Simulation & Gaming, 33(4), pp. 441–467, (2002)

Gestwicki, P., Sun, F.-S.: Teaching Design Patterns Through Computer Game Development. In: ACM Journal on Educational Resources in Computing, 8(1), pp. 1–22, (2008)

Girard, C., Escalle, J., Magnan, A.: Serious games as new educational tools: how effective are they? A meta-analysis of recent studies. J. of Computer Assisted Learning (2012)

Greitzer, F. L., Kuchar, O. A., Huston, K.: Cognitive science implications for enhancing training effectiveness in a serious gaming context. J. Educ. Resour. Comput., 7(3), art. 2, (2007)

Gun-Tactyx, `http://apocalyx.sourceforge.net/guntactyx/`, accessed 19 January 2011

Hartness, K.:Robocode: using games to teach artificial intelligence. J. of Computing Sciences in Colleges, 19(4), pp. 287–291, (2004)

Jenkins, T.: On the Difficulty of Learning to Program. In: 3rd annual Conference of LTSN-ICS, pp. 53–58, (2002)

Johnson, W., Wu, S.: Assessing Aptitude for Learning with a Serious Game for Foreign Language and Culture. In: Intelligent Tutoring Systems, pp. 520–529, (2008)

Kelleher, C., Cosgrove, D., Culyba, D., Forlines, C., Pratt, J., Pausch, R.: Alice2: Programming without Syntax Errors. In: 15th annual symposium on the User Interface Software and Technology, (2002)

Kernel Panic, `http://springrts.com/wiki/Kernel_Panic`, accessed 19 January 2011

Kiili, K.: Foundation for Problem-Based Gaming. In: British Journal of Educational Technology, 38(3), pp. 394–404, (2007)

Kinnunen, P., Malmi, L.: Why students drop out CS1 course?. In Proceedings of the second international workshop on Computing education research, pp.97–108, (2006)

Lane, H. C., Hays, M. J., Auerbach, D., Core, M. G.: Investigating the relationship between presence and learning in a serious game. In: Intelligent Tutoring Systems, pp. 274–284, (2010)

Lee, S. Y., Mott, B. W., Lester, J. C.: Optimizing Story-Based Learning: An Investigation of Student Narrative Profiles. In: Intelligent Tutoring Systems, pp. 155–165, (2010)

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., Resnick, M.: Scratch: A Sneak Preview. In: 2nd International Conference on Creating Connecting, and Collaborating through Computing, pp. 104–109, (2004)

Marfisi-Schottman, I., George, S., Tarpin-Bernard, F.: Tools and Methods for Efficiently Designing Serious Games. In 4th Europeen Conference on Games Based Learning ECGBL2010, Copenhagen, Denmark, 21-22 October, pp. 226-234, (2010).

Marvin's Arena, `http://www.marvinsarena.com/`, accessed 19 January 2011

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B.-D., Laxer, C., Thomas, L., Utting, I., Wilusz, T.: A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. Working group reports from ITiCSE on Innovation and technology in computer science education, pp. 125–180, (2001) Muratet, M., Torguet, P., Jessel, J.-P., Viallet, F.: Towards a serious game to help students learn computer programming. In: Int. J. Comput. Games Technol., vol. 2009, pp. 1–12, (2009)

Muratet, M., Torguet, P., Viallet, F., Jessel, J.-P.: Experimental feedback on Prog&Play: a serious game for programming practice. In: Computer Graphics Forum, 30(1), pp. 61–73, (2011)

Muratet, M., Delozanne, E., Torguet, P., Viallet, F.: Serious game and students' learning motivation: effect of context using Prog&Play. In: Intelligent Tutoring Systems, to appear (short paper), (2012)

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., Paterson, J.: A survey of literature on the teaching of introductory programming. In: SIGCSE Bull., 39(4), pp. 204–223, (2007)

Robot Battle, `http://www.robotbattle.com/`, accessed 19 January 2011

Rodrigo, M. M. T. et al.: Comparing Learners' Affect While Using an Intelligent Tutoring System and a Simulation Problem Solving Game. In: Intelligent Tutoring Systems, pp. 40–49, (2008)

Rowe, J. P., Shores, L. R., Mott, B. W., Lester, J. C.: Integrating Learning and Engagement in Narrative-Centered Learning Environments. In: Intelligent Tutoring Systems, pp. 166–177, (2010)

Thomas, P., Yessad, A., Labat, J.-M.: Petri nets and ontologies: tools for the "learning player" assessment in serious games. In: Advanced Learning Technologies, pp. 415–419, (2011)

Yessad, A., Labat, J.-M., Kermorvant, F.: SeGAE: A Serious Game Authoring Environment. In: Advanced Learning Technologies, pp. 538–540, (2010)

Zyda, M.: From Visual Simulation to Virtual Reality to Games. IEEE Computer, 38(9), pp. 25–32, (2005)